10/505151
DT15 Rec'd PCT/PTO 20 AUG 2004

DATA BLOCK STORING METHOD IN A MEMORY

This invention relates to the field of data storage in a re-writable digital memory of semiconductors that keeps its contents in cases where the power supply is interrupted. More particularly, the invention relates to the management of the memory space available by means of a storage method for data blocks in the memory.

The semiconductor memories are used in all the applications that comprise microprocessors, for which it is necessary to store the program and the necessary data for their functioning.

The data are introduced, in general, in the memory in predetermined addresses, namely, defined during the development of the program, or sequentially, namely, by successive blocks following the blocks already present in the memory. Likewise these blocks can be re-recorded on other blocks that are already present, in order to renew the data that have become obsolete. A block is a sequence of bits or bytes of predetermined length or size that includes a header containing a block identifier and a number defining its length.

According to the instructions of the program, the data are stored in the memory in positions defined by addresses. The latter are determined by means of parameters contained in the program. These reserved positions are situated in any area of the memory whose limits are defined by a field of addresses. This range, determined in this way, corresponds to the capacity available that is, in general, greater than the maximum quantity of data that can be stored in it.

Numerous applications of more and more sophisticated data processing are installed on smaller and smaller physical mediums. Therefore, the capacity of the memories used for microprocessors must be optimized to the maximum. These cases appear for example in different modules of electronics like smart cards or in any other medium that includes numerical processing components of miniaturized data.

Some applications, particularly access control ones, user identification ones or those of electronic payment, must respond to more and more safety demands in order to avoid fraud. Indeed, the functions of a card can be revealed after deep analysis of

the contents of the memory associated to the processor. For example, the debit mechanism of a payment card produces a set of data that are stored in positions of the memory, which are predetermined by the program. For each operation done by the card, a well-defined configuration of the data in the memory corresponds to it. This situation leaves a door open to piracy of the cards whose functionalities can be copied or simulated onto other cards.

The object of the present invention is to propose a protected storage method for data in a memory so as to avoid falsifications of their contents by analysis. Another object consists in limiting the wearing of the memory by means of improved management of the data reading/writing cycles.

This objective is achieved thanks to a storage method of a plurality of data blocks in a digital re-writable memory of semiconductors controlled by a memory manager and characterized by the following steps that consist of:

- randomly determining an available area,

- storing the data block in the area chosen in this way.

By "available area" one understands an area of the memory that is free of data or that contains data replaceable by new ones like in case of an updating for example.

The method according to the invention allows the storage of data blocks in positions of the memory that are always different although the program carries out a series of identical operations. For example, a 10 units debit operation on a card will not have the same effect on the memory contents structure at each execution of the same debit function. Furthermore, two identical cards that carry out an identical operation will have a completely different structure in the contents of their memory. In this way, an analysis of the data of a card will not allow one to reproduce an image of the operations of the first card with the other and vice versa.

Besides the aspect relative to security, the method of the invention allows, thanks to the reading/writing in randomly chosen areas, better distribution of the wearing of the memory. Therefore there will not be areas in the memory that are worn out more quickly than others, like when numerous data reading/writing cycles are carried out always in an assigned place of the memory.

The random selection of an available memory area can be carried out according to different variants:

1- The result obtained after the exploration of the memory constitutes a list of addresses corresponding to the available areas. This list is kept temporarily in a random access memory. Afterwards an address is randomly chosen from this list, and then the data block is stored in the area of the memory indicated by this address. A variant of this method consists in continually maintaining a table with the available areas and randomly choosing an address among them.

2- Exploration of the memory determines the maximum number of available areas. A random selection of a number n between 1 and the number of areas found designates the area where the block must be stored. For example, there are 20 areas available, the random selection of a number between 1 and 20 gives 8, the block is therefore stored in the eighth available area.

3- A number N is randomly determined between 1 and the maximum number of areas possible. The memory manager sequentially searches said N^{th} available area, and if it reaches the end of the memory before finding this area, the memory manager restarts the search from the beginning of the memory until the N^{th} available area is found.

The invention will be better understood thanks to the following detailed description that relates to the attached figures given as a non-limiting example, that are:

- Figure 1 shows the storage of some data blocks with the same length in a portion of the memory.

- Figure 2 shows the storage of variable length blocks.

- Figure 3 illustrates the storage of blocks taking a predetermined gap into account.

Figure 1 illustrates a case in which the data blocks all have the same length l. They are memorized randomly in available areas whose length corresponds to a multiple of the block length to be memorized. For example, if the blocks all have a length of 10 bytes, they can be distributed at random in positions of 10, 20, 30, 40, etc. bytes. The available area can be bigger than the block to be memorized. For example, a

block of 10 bytes B8 can be placed in a space e2 of 30 bytes and with an offset of 20 bytes with regard to the beginning of the available area, namely at 20 bytes from the preceding block B5.

During the storage of a new block Bn, according to the first variant of the invention, the memory manager will explore the memory and will deduce the available addresses from there e1, e2, 1, e2, 2, e3 and e4 understanding that space e2 allows storing two blocks of fixed length. Once these addresses are determined, a random variable can be used to define the address of the available area where block Bn will be stored.

According to the second variant, the manager finds 5 available areas whose length corresponds to those of the blocks to be stored. A random selection of a number between 1 and 5 gives 3, the block Bn will therefore be stored in the third area, namely, in e2, 2.

According to the third variant, the maximum number of available areas Z is 13. The manager randomly determines a number N between 1 and 13, for example 8, afterwards it explores the memory to find the eighth available area. A first run reveals that there are 5 available areas and a second run from the beginning determines that position e2, 2 (the third) corresponds to the eighth area. In brief, if the determined random number N is greater than the number of available places P, the position of the free space is defined by the random number N modulo the number of available places P. Here, in the example, N=8 is bigger than P=5, so the block will be stored in position 8 modulo 5 = 3rd place. In the particular case where N modulo P is equal to 0, the block can be situated in the first or last position. According to another variant, the random number N can be defined again until obtaining a value N modulo P different from zero.

Figure 2 a) represents the case in which the blocks have a variable length and are separated or not by free areas. For example a block B2 of 20 bytes begins at 5 bytes from the preceding block and ends 5 bytes before block B4. The areas or free spaces e1 and e2 before and after B2 can be occupied if B2 and B4 must be replaced, for example. It is the same for all the other free spaces that are either

occupied or that move during the storage of new blocks Bn instead of the preceding ones.

A new block Bn can be stored in the remaining free spaces or substitute one or several of the blocks still present that are no longer useful. In this way the freed space allows the storage of several smaller blocks or a bigger block that occupies all or part of the space. Figures b) and c) show an example of updating: a new block B12 has been stored in the free space e4. Block B10 is replaced by a bigger block B11 that, therefore, occupies all the freed space e9 between B7 and B9. Blocks B2 and B4 have been replaced by B13 that occupies half of the freed space e10. The new free space e11 created in this way will be used during the next storage of blocks.

According to another variant of the invention illustrated in figure 3, the program determines a usual length m of the data blocks to memorize. This value can correspond to the most frequent length of blocks or, in certain cases, to the average length of the blocks. After random selection of the available storage area, the block will be memorized either directly after an already present block, in the case where the block has an equal or longer length than said length m, or with an offset of n bytes in order that the length of the block and the offset n is equal to the length m. This variant allows, after the deletion of this block, freeing a space that will be used very quickly. Without this offset foreseen at the time of storage, the position freed by this block will have very little chance to be used again.

According to our example, the normal length m of the blocks is 15 bytes; the blocks have lengths that vary between 5 and 20 bytes. Two cases are shown:

If the length of the block Bn to be stored is smaller than the current length m, Bn is stored at a pitch m starting from the preceding block in order to leave a free space equal to the difference between m and the length of Bn. According to the example above, a block of 10 bytes is placed at 15-10 = 5 bytes from the previous block. Figure 3 a) shows blocks separated by available areas. In figure 3 b) a block B6 is stored in the free space e2, the length of B6 being smaller than the current length m, B6 is placed at a pitch m starting from the preceding block B2. The space e5

between B2 and B6 is equivalent to the difference of length between m and the length of B6.

If the length of the block Bn to be stored is greater or equal to the current length m, Bn is placed immediately after the preceding block. In figure 3 b) B7 is bigger that the value m and is placed therefore in e4 after B5 without leaving any free space between them.

The method according to the invention can also be applied to more important memories that have a structure in form of a table or matrix that allows direct access to the data blocks. In such a case some pointers define the positions available in the memory. The latter are chosen randomly before storage of the data blocks in the memory.

The data from which the blocks have been stored according to the method of the invention can be reconstructed by analysis, either the identifiers contained in the headings of the blocks or the addresses of each block contained in a previously memorized table.

In one embodiment of the invention, the table that contains the direct access pointers is contained in a second secured memory. In this way, it is possible that the main memory is an unsecured such as a memory of a computer and that the pointer table is stored in a security module (a smart card or similar element). Each data block comprises an identifier that will be transmitted to the card eventually along with the size of the data. In exchange, the card randomly determines a pointer among the free pointers as described previously and returns this pointer to the host computer. In parallel, the card stores the data identifier along with the pointer value.

It is noted that it is possible to avoid storing the identifier with the data block in the main memory, this information being found only in the secured memory. Storing the blocks without their identifier in the main memory will then prevent any identification of these blocks through an analysis of the memory.

In the case of reading, the identifier is transmitted to the card that searches for the corresponding pointer in its secured memory; a pointer that will be returned to the host computer to accede to the data blocks in the main memory.

In this way, each main memory content is unique and cannot be transported from one computer to another. It must compulsorily be accompanied by the security element that stores the pointer table.